The Co-op Spirit

PEOPLE HELPING PEOPLE WITH PERSONAL TECHNOLOGY *August 1991*

DOS 5.0

Inside:

A Letter from the President

APPLE //

A Skeleton Desktop Application

MS-DOS

Micro Review of DOS 5.0

Bits and Bytes

Programming and Visual Basic

a publication of **TechAlliance**

Volume 1, Number 7 August 1991

Published by

TechAlliance

19309 – 68th Avenue South, Suite R-109
Kent, Washington 98032
Office: 206/251-5222
FAX: 206/251-6190

THE CO-OP SPIRIT is the monthly newsletter of TechAlliance, a non-profit computer cooperative committed to providing members with resources that increase the understanding and enjoyment of personal computers. The co-op provides members with user groups for the Amiga, Apple // Family, Macintosh, and MS-DOS machines. Both beginning and advanced assistance are provided at our various Special Interest Group Meetings. The Kent, Washington office is presently being set up as a resource center for our local members. Members can peruse the software and book library. This newsletter is our official organ and is sent to every member. It consists mostly of contributions from the membership which include industry news, machine-specific articles, and other items of interest.

MEMBERSHIPS: TechAlliance Memberships may be obtained by filling out the form on the back cover of this newsletter and mailing it, along with your dues, to TechAlliance. Annual Membership dues are $25.00 per year; $35 per year outside the U.S.

DISCLAIMER: The editor, staff, and contributors of this newsletter assume no liability for damages resulting out of the publication of any article, hint, or programming code. In addition, the opinions presented regarding reviews and/or the mention of products indicate no affiliation or endorsement of products by TechAlliance. The views expressed in this newsletter are those of individual contributors and not necessarily those of TechAlliance.

PERMISSION TO REPRINT: Unless specified otherwise, articles in this publication may be reprinted by other nonprofit user groups, provided credit is given in the byline to both the author and newsletter from which it was taken, and a copy of the user group's newsletter is mailed to TechAlliance.

SUBMISSION DEADLINES: The deadline for articles, news, and meeting announcements will be the 1st of the prior month. All articles to be submitted on either a 3 1/2" or 5 1/4 " disk in ASCII format with a hardcopy included, or articles may also be uploaded in ASCII format to the TechAlliance BBS with a hardcopy submitted by mail. Be sure to indicate on what type of machine the disk was formated. TechAlliance reserves the right to review, edit, delete or accept articles at its discretion.

MEETING SCHEDULE: Please see last pages of newsletter for meeting schedules and times.

Editors ..... Sharon Farquhar, Director
Assistant Editor ..... Norm Dodge
Design/Desktop Publishing ..... Arnon Loawhakasetr 223-4221

TechAlliance Board of Directors

John Keithly, President — *Dave DeGroot, Treasurer*
Anthony Fortini, Vice-President — *James Walley, Secretary*
Sharon Farquhar, Director — *Robert Huelsdonk, Director*
Richard Hubert, Director

SIG Groups	BBS SIGOPS	SIG Leaders
Amiga	John Keithly	Peter Barnes
Apple//	Richard Hall	Phil Lindberg
Apple //GS	Gerry Drake	Phil Lindberg
Macintosh	Dave Showalter	Dave Maulding
MS-DOS	Bob Caldwell	Sharon Farquhar
Mac Programmer	Robert Riebman	Robert Riebman
//GS Programmer	Robert Hess	
"C" Programmer		Doug Trueman

TechAlliance BBS System Operator – Paul Stout
BBS: 206/251-0543 (8 bit, no parity, 1 stop bit)

From the Editor's Desk

In the July issue of The Co-op Spirit, the byline on the article "The First 20 Things to Do and Notice with System Software 7.0" was inadvertently left off. We believe in giving credit, where credit is due. The authors of that article, which was reprinted from the Kansas City MacCore, June 1991 issue, were Guy Kawasaki and Mike McCannon. Sincere apologies are extended for this error.

LETTERS.....

Take the opportunity to comment on articles in your newsletter and make suggestions to your organization.

The following letter was received in the TechAlliance Office this month:

I would like to express my irritation at a couple of things in the June 1991 issue.

The first is the requirement that articles must be submitted on 3.5" disk with no specification of format. Many people with Apple II's do not have such a drive. Presumably with the right combination of equipment you can read Mac, MS-DOS, and Apple II 3.5 inch disks, but it must be fun figuring out what you've got and I wonder at the deep satisfaction at cutting out the older machine owners.

Art Radcliff's article on mastering the oldest Apple II even with its "overwhelming . . . simplicity" should have had a least a couple of cautions attached to it, which I would like to state here.

When he says he explored the potential of PC machines and found them "extremely awkward in switching between BASIC and Assembly routines" he is making a statement similar to: "When automobiles came along, I tried getting off of one while it was moving, as I often do a horse when I need to pickup something on the ground, but it was extremely awkward." Modern QuickBASIC for the MS-DOS machines has a very smooth interface to Assembly language which is virtually unneeded due to the powerful drawing commands included in the language, which allow virtual mapping of the screen, direct coordinates, and many other features which I know from experience are major programming task on the II.

Further, he is misleading your readers by suggesting that the few colors available on the II+ are adequate for anything beyond crude cartoons or rude abstracts, made even more exciting by the interactions between adjacent bits. And working with the odd screen memory layout of the Apple II+ in assembly language (which I have done since 1979, so I am not bitter about a failure in my past) is something to turn off all but the most dedicated beginner.

At one time, Call A.P.P.L.E. was a necessity nation-wide for Apple II programmers. Now I find it difficult to consider recommending what I am seeing to anyone. Such a come down.

Mike Firth
Dallas, Tx

Mike, at this point TechAlliance is an all volunteer organization. Perhaps had you known that, the tone of your letter would have been more friendly toward those volunteers. As a volunteer editor of the newsletter (and a very inexperienced one at that), I must beg for reprieve from some of your comments. First of all, in reference to the requirement that disks be 3 1/2", the 5 1/4" disks we have received, have been virtually unusable

Continue to page 3

Lucky Number Seven! "From the President"

Yes, this is the seventh (and hopefully for you, the best yet), issue of the Co-Op Spirit. And yes, despite trials and tribulations both financial and legal, we have persevered and intend to continue as "people helping people with computer technology" long into the future. We particularly strive to support computer users of the Apple II+/C/E/GS, Macintosh, IBM & compatibles, & Amiga.

Howdy, I'm your new president. I was elected to the board by you the membership in March, and was thereafter elected in June by the board of directors to become the president of TechAlliance. For those of you who might not recall, my comments as a candidate on the election ballot centered around balancing the TechAlliance budget.

The board has decided to go to an all volunteer staff (as of 1 August). TechAlliance will then come very close to attaining a desirable cash flow position. It may still be necessary to cut costs pertaining to our local meeting site, but our newsletter and BBS expenses should be covered.

I'm especially pleased to report that the involuntary bankruptcy claim that was directed at TechAlliance this past spring, has been dropped in an out of court settlement favorable to us. This action gives me considerable optimism that we can now expect to soon clear ourselves from the morass of legal entanglements which have beset TechAlliance for nearly the past year.

Anything else to dispel those doom and gloom rumors I've been hearing? Yes, most definitely! As some of you know who use our BBS, the board of directors posted a message last month asking for ideas that could improve our service to the existing membership—and of course—encourage new memberships.

In response, Robert Riebman, our Mac Programmer's SIG leader, has volunteered to set up, on an experimental basis, a Usenet feed on the TechAlliance BBS which will be devoted to current coverage of the Macintosh. Bob hopes to update the Usenet message base every three days or so. If there is sufficient favorable response to this effort and more volunteer support becomes available, the Usenet feed will be expanded to cover other computers and special interests groups (SIGs).

Also as a result of BBS feedback, especially from Syd Schofield, our resident LOGO Guru, there is now a new TechAlliance BBS SIG devoted to "Education et al". It too contains traffic from Usenet (albeit for now only a few choice messages per day).

Again, if you the BBS members are inclined to respond favorably, our Usenet coverage can also be expanded from read only status to include e-mail access via FidoNET. FidoNET itself should be installed "real soon now" once our BBS memory board problems are cured. In keeping with our desire to get full membership participation, the board of directors has asked me to request from this newsletter audience whether you might have additional suggestions for bringing back the regrowth of our Co-op. Yes, I know we need to publish our membership directory and consultant catalog—these will be included in the next issue of the Co-op Spirit and for the 20% of you using the BBS, these databases will be posted there as well.

So what else are you looking for? Okay, let's be candid. I know there are many of you who would like to see the return of CALL A.P.P.L.E. So would I. My wife is still using my Apple II+ (clone) in her first grade classroom. But CALL A.P.P.L.E. has been gone for almost two years now and it's not likely that many of the authors who contributed so well and so long could be persuaded or entreated to come back from whatever magazines they have migrated to. (I would be delighted to be proven wrong here.) Besides, there are many of you that have migrated to other machines. We have done a preliminary survey and have found that about 60% of our members now are Macintosh users, 40% Apple II (of one kind or more), 40% are IBM/clone qualified, and about 25% use Amigas, Unix workstations, and main frames. So how can we carry on by still providing service worth your membership fee?

What kinds of information can we provide to you that's lacking now? Please take some time to consider this question and give us your preferences. I read ALL your mail and I would be equally happy to take your Fax (206) 251-6190 traffic or public/private BBS message (206) 251-0543. Even phone call answering machine messages are fine (206) 251-5222.

Okay, I think it's time to wait for your reply—and in the mean time there is always a lot of work that needs tending to, like reorganizing the TechAlliance software libraries . . . Hmmm, that's an idea already. . . Anymore?

Very Sincerely,

John Keithly

Editor's Note: The percentages in the above letter do not equal 100 percent because many members use more than one machine type, thus throwing the percentage off.

Continue from page 2...Editor's Desk

thanks to the to the postal service. Therefore, the most desirable methods of sending articles are either to upload the article to the TechAlliance BBS, with a hard copy mailed, or a disk properly packaged and mailed. Because of durablity, the 3 1/2" disk was very desirable. If the articles received are not readable by my machines, then I must call on a little help from other volunteers. And what a JOY it would be, if members like yourself submitted enough articles for this newsletter so that I would have to figure out what machine format I have received. But the points are well taken, and the submission requirements will be changed. Finally, there was no attempt at cutting out the older machine owners. In fact, I would be grateful to have them participate as volunteers to make this newsletter and organization the very best possible. The purpose of this newsletter is to provide information to all users of Apple, Amiga, Macintosh, and MS-DOS machines, and no age restrictions apply..

As a member of TechAlliance, perhaps YOU and others can help make "The Co-op Spirit" as valuable as "Call A.P.P.L.E." once was to our readers! So please continue sending letters, but also realize a co-operative is what the members themselves make of it.

Editor

A Skeleton Desktop Application:

Window Management, Part 4

by Sandy Mossberg

Every application worth its salt must have a window or two to let the little guy who lives inside send out a message or receive a message. The Skeleton Desktop Application is no exception so in this installment, window dressing is added. It sounds like a sitcom. I hope it doesn't read like one.

A window is a rectangular area which displays information on the Desktop. Alert windows (e.g. our About box), alert boxes, and modal dialog boxes (e.g. our shut down box) are kinds of windows which wrest control from TaskMaster and thus operate in a modal state. System windows are handled jointly by the Desk Manager and the application in a modeless fashion. Modeless dialog boxes and standard application windows are the remaining window categories. Both function modelessly under control of the application. The subsequent pages are devoted to incorporating these latter two window types into the Skeleton.

Revisions to SKELETON.ASM

Before windowing code can be created, alterations must be made to the SKELETON.ASM file. Fairly extensive revisions are needed in the event loop and in the handler which closes windows. Other modules demand less modification. Turn your attention to Listing 1 and let us proceed in numeric sequence.

(1) Global program data: Add two variables to the Globals module. TMCode holds the original code returned by TaskMaster and MLPtr contains the pointer to the modeless dialog's GrafPort. If MLPtr is zero, the modeless box is not open.

(2) Desktop initialization data: In the data area of InitDesktop, replace the old Samples menu with the new one provided. The names of the first two items have changed and these options appear undimmed in the menu.

(3) Initialize application specific data: By replacing the prior InitProgram module with the given code, you are signaling that the modeless dialog box is closed when the program is started up. As before, the quit flag initially is turned off.

(4-5) Event loop code and data: A standard application window requires minimal overhead, whereas a modeless dialog calls for substantial changes to the TaskMaster event loop. Replacement EventLoop code is presented (4) even though half of it remains unchanged. In the data section (5), substitute DoMLEvent for DoNothing in task table event number 12 and add the table called MLFlagTbl.

Here is how the new event loop works:

All possible event codes returned by TaskMaster are stored in EventType and TMCode, a local and global variable, respectively. While TMCode remains intact for use by the modeless dialog handler, EventType may be modified by the event loop.

IsDialogEvent examines the What field of the task record to determine whether an event pertains to a modeless dialog box. Input for this function is a pointer to the event record or task record, and the call passes back a Boolean word. If the result is false (i.e. zero), a dialog related action did not take place and the event loop continues along its merry way. Upon detecting a result of true (i.e. not zero), a modeless dialog event occurred, EventType is remapped to activate task number 12, and the application's modeless dialog handler (DoMLEvent) is accessed via TaskMaster's jump table. More about DoMLEvent in a while.

Not all events are passed along to IsDialogEvent. MLFlagTbl holds a flag for each potential event in the task table. If the flag is on (i.e. one), IsDialogEvent analyzes the event. Should the flag be off (i.e. zero), the corresponding activity is bypassed. Our application allows IsDialogEvent to process mouse-down events, mouse-up events, auto-key events, update events, activate events, events within the content portion of the dialog, and special menu events. An application-defined event does not require consideration because it is mapped later.

IsDialogEvent becomes schizophrenic when it encounters a special menu option (IDs 250-255). If a keyboard equivalent is entered, a true result is returned and control is routed to the modeless dialog handler. When the mouse is used to select a special menu item, however, IsDialogEvent detects a mouse-down event outside of the dialog box and returns a false value. Therefore, DoMLEvent never "sees" the event. This latter situation could be remedied by adding code which (1) employs FrontWindow to obtain the pointer to the active window, (2) compares this pointer to MLPtr, and (3) changes EventType to 12 if the pointers match and TaskMaster returned wInSpecial (code number 25). The actual editing code would then be processed within DoMLEvent. If you wish to carry modeless editing to its logical conclusion, which this article cannot do, obtain the Apple IIgs Source Code Sampler disks from APDA. Be sure the version is later than 1.0.1. Jim Luther of Apple II Developer Technical Support has promised to provide editing capabilities for modeless dialogs in the DLOG.AII source code.

(6) Menu handler data: Change the handlers for Items 261 and 262 from DoNothing to DoWindow1 and DoWindow2, respectively.

(7) Closing the front window: When a window is open, its menu option is dimmed to indicate that the window is already on the Desktop. On closure, the menu option must be undimmed so the window can be reopened. Whenever our application opens a window, the menu ID number of the selection is tucked into the reference constant field of the window record. These manipulations will become more clear when we get to the actual creation of windows, but the information is necessary to understand the DoClose replacement code explained below.

A pointer to the topmost window is obtained by FrontWindow. If the pointer is nil, no window is open and the subroutine returns to its caller. The empirical assumption of an active system window is confirmed by the carry flag following the CloseNDAbyWinPtr call. On carry clear, a New Desk Accessory (NDA) window was closed and we are finished. On carry set, an application window is open.

Before distinguishing between a standard window and a modeless dialog, the

menu ID number is extracted from the reference constant field by GetWRefCon then the menu option is undimmed using EnableMItem. If the pointer to the front window matches the modeless dialog's pointer, the dialog box is closed via CloseDialog and MLPtr is zeroed. If the pointers are not equal, a standard application window must be active, in which case CloseWindow performs its magic.

(8) Incorporate windowing code into application: The ending COPY command copies Listing 2 into SKELETON.ASM.

Revisions to NULL.ASM

The null event handler must accommodate the edit line function of the Skeleton's new modeless dialog window. Although the dialog has not yet been presented, the modifications noted below should be easy to comprehend.

(1) Blink edit line insertion point in modeless dialog: As you will see shortly, an edit line box contains a blinking vertical bar which represents the insertion point for text. For the cursor to blink continuously, the modeless dialog handler must be invoked with each null event. A call to DoMLEvent does the trick.

(2) Update status of Edit menu items: Although replacement code is given, the only change involves the activation of Edit menu options when the modeless dialog box is active. By comparing the modeless pointer with the front window pointer, you can determine if a modeless dialog is on top.

A Standard Window

The DoWindow1 module in Listing 2 creates a full-featured standard application window. The working code is short and sweet. When a menu item is selected, the task record's TaskData field holds the menu item ID and menu ID in its low and high order words, respectively. The item ID number is saved in the window's parameter list for use by DoClose which reenables the menu option using this data.

NewWindow is the Window Manager's big gun. Its single input is the address of the window's parameter list and NewWindow returns a pointer to the window record whose composition is detailed in the Apple IIgs Toolbox Reference: Volume 2. After the window has been created, the Window 1 menu option is dimmed by DisableMItem.

A description of NewWindow's parameter list is given in Table 25-9 of the Apple IIgs Toolbox Reference: Volume 2. The following fields demand special attention:

wFrameBits consists of flags which define attributes of the window and its frame. Figure 25-19 and the accompanying text in Volume 2 of the Apple IIgs Toolbox Reference show the meaning of each bit. Set bits in the Skeleton's wFrameBits indicate the following characteristics of Window 1:

Bit 2: Allocation of window record by NewWindow
Bit 5: Visibility
Bit 7: Dragging by title bar
Bit 8: Zoom box
Bit 10: Grow box
Bit 11: Horizontal scroll bars
Bit 12: Vertical scroll bars
Bit 14: Close box
Bit 15: Title Bar

wRefCon is utilized by the Skeleton to store the menu ID number for the purpose of undimming the menu item when the window is closed.

wColor defaults to a white title and black title bar, frame and controls. If you want racing stripes on the title bar, a color table is necessary (see Figure 25-9 in the Apple IIgs Toolbox Reference: Volume 2.)

Arrow and page scrolling should be coordinated with the size of the data area so that the desired amount of information is displaced with each click on the control. Experimenting with these values is unavoidable.

wContDefProc points to the content definition procedure called by TaskMaster in response to an update event. If the pointer is nil, the application must update its window. Because the data bank occupied by TaskMaster code may not coincide with that of the definition procedure, the data bank should be saved on entry, then equated to the program bank, and restored on exit.

Wind1Update is the Skeleton's feeble notion of drawing in Window 1. Starting at the position established by MoveTo, text is placed on the screen by DrawString. You see the text only if the visible region of the window contains the area to which the data is drawn. Although having TaskMaster manage the details is convenient, more rapid action can be accomplished by a custom update procedure which is beyond the realm of a skeletal application.

A Modeless Dialog Window

A modeless dialog offers the features of a dialog box in an application type window. To support a modeless dialog, an application must contain code segments which perform the following functions:

(1) Detecting dialog-related activity within the main event loop.

(2) Creating the dialog window.

(3) Handling events within the dialog window.

The first routine has been described earlier. The other two sections (see Listing 2) are presented below.

Creating a Modeless Dialog

DoWindow2 utilizes NewModelessDialog to create an empty dialog box. The following input parameters are passed:

(1) A pointer to the global coordinates of the window frame.

(2) A pointer to the window's title string.

(3) The plane of the window (identical to the wPlane parameter for NewWindow).

(4) Frame flags describing the window's characteristics (identical to the wFrameBits parameter for NewWindow).

(5) An application defined reference constant (identical to the wRefCon parameter for NewWindow). The Skeleton stores the menu ID number for Window 2 in this cubbyhole.

(6) A pointer to coordinates of the zoomed window. Since our window has no zoom box, the pointer is nil.

NewModelessDialog returns a pointer to the dialog's GrafPort. Once the window is created, items may be added using NewDItem or GetNewDItem. Being a template freak, I opted for the latter function. Item templates for modeless dialogs are akin to those for modal dialogs and may be reviewed in Part 3 of this series. Our modeless dialog contains two buttons, static text, and an edit line box.

Handling Activity Within a Modeless Dialog

DoMLEvent is accessed when the event loop detects unfiltered activity within the modeless window. If the dialog box is closed, the modeless handler returns forthwith. Because special menu events are passed to IsDialogEvent, DoMLEvent is responsible for handling these events. Thus, if TaskMaster returns wInSpecial, highlighting is removed from the title of the File menu or Edit menu and the extended task event record is examined for the ID number of the menu item selected. If Close was chosen, a jump to DoClose closes the modeless box

and returns to the caller.

DialogSelect is the engine which drives a modeless dialog. Input consists of pointers to the task record, to a location which stores a pointer to the dialog's GrafPort, and to a location which houses the ID number of the item hit. If the event involves an enabled dialog item, DialogSelect returns a Boolean true result and places the item's ID number in ItemHit. Generally, a false value is returned if no action is required. Peculiarities of DialogSelect are detailed in the Apple IIgs Toolbox Reference: Volume 1.

DoMLEvent is also called by the null event handler so that the cursor in the active edit line box will blink via the invocation of LEIdle by DialogSelect.

Although our modeless dialog box contains simple controls, we can still have fun with it. The window can be closed and dragged but not resized. Clicking on either button produces a beep. Type your name on the edit line. The insertion point is marked by a blinking vertical bar. Text is selected by dragging the cursor across one or more characters, words are highlighted by double-clicking, and all text is selected by triple-clicking. The SHIFT key extends the selection range, the DELETE key erases the character behind the insertion point, CONTROL-X erases the entire line, and CONTROL-Y deletes text to the right of the blinking bar. Cut, Copy and Paste functions are available on selected text by using keyboard equivalents. The old mouse-in-the-menu trick does not work. You will be amazed at the amount of editing which can be performed in the absence of specific program code. I tip my hat to DialogSelect and TaskMaster, the two functions which make this possible.

Final Thoughts

Skeleton Desktop Application is now a mature Skeleton, if such an entity exists. In the final installment, help for disabled menu items will be provided and a new menu option will enable you to change the boot slot. We are on the home stretch.

In order to create the macro file use the command:

```
MACGEN SKELETON.ASM
SKELETON.MAC 2/AINCLUDE/M=
```

In order to assemble use the command:

```
ASML SKELETON.ASM
FILETYPE SKELETON S16
SKELETON
Listing 1
```

```
**********************************************
*
* SKELETON.ASM source listing (Part 4)
* by Sandy Mossberg
*
* APW Assembler
*
*   Note lines with *** are added in this article
*
**********************************************
        mcopy   skeleton.mac
        keep skeleton

DPHndl  gequ    $00     ;handle to DP
DPAddr  gequ    $04     ;DP address
MenuPtr gequ    $08     ;ptr to Menu Mgr
ScreenMode      gequ    $80     ;640x200 mode
ScreenWidth     gequ    640     ;640 screen width

**********************************************
*
MainLoop start
*
* Main program loop:
*
**********************************************
        using   Globals

        phk
        plb
        jsr     StartTools
        bcs     QuitPgm ;pgm aborted
        jsr     ;InitDesktop
        jsr     ;InitProgram
        jsr     EventLoop
        jsr     ShutTools
QuitPgm _QuitGS QuitParm
        end

**********************************************
*
Globals data
*
* Global program data:
*
**********************************************
MyID    ds      2       ;master memory ID
QuitFlag ds     2       ;quit flag
FrontWinPtr     ds      4       ;front window ptr
ModalPtr ds     4       ;modal dialog ptr
RadioStatus     ds      2       ;active radio ID
TMCode  ds      2       ;TaskMaster code ;***
MLPtr   ds      4       ;ML dialog ptr  ;***

QuitParm anop   ;Quit parmlist:
        dc      i2'2'   ; class 1 GS/OS
        dc      i4'0'
        dc      i2'0'

TaskRec anop    ;task record
What    ds      2
Message ds      4
When    ds      4
Where   ds      4
Modifiers ds    2
TaskData ds     4
TaskMask        dc      i4'$FFFF' ;handle all events
        end

**********************************************
*
InitDesktop start
*
* Initialize desktop:
*
**********************************************

        PushLong        #0      ;draw whole screen
        _RefreshDesktop

        ldx     #MTIndex
MenuLoop phx                    ;save tbl index
        PushLong        #0      ;result space
        lda     MenuTable+2,x
        pha                     ;ptr to template
        lda     MenuTable,x
        pha
        _NewMenu        ;get menu space

        lda     1,s     ;check handle
        ora     3,s
        bne     ML01    ;got handle
        sec             ;bombsville
        jmp     CheckErr

ML0 PushWord    #0      ;front of list
        _InsertMenu ;insert menu
        plx ;restore tbl index
        dex ;index next entry
        dex
        dex
        dex
        bpl     MenuLoop

        PushWord #1 menu ID
        _FixAppleMenu   ;add NDAs

        pha
        _FixMenuBar ;get menu bar size
        pla             ;height not needed
```

Continue to page 7

Programming and Visual Basic

by George Palo

Why program? Why Visual Basic? The answer to both these questions is simple. It's fun! In the MS-DOS world there seems to be many PC owners who just acquire. They acquire gigabyte hard drives, super fast processors, megabytes of ram, and a zillion shareware and commercial programs. These "power users" I'm sure are having tons of fun, but they're missing at least 50% of the REAL fun.

The first program I wrote was in 1967 on a terminal in our office in Seattle, which was connected to a timeshare computer at Stanford University. In those days we didn't have monitors, we used teletype. When the old teletype, after what seemed like hours, came back and typed "Hi George", I was hooked. Since then I've owned a TI99/44a, a Commodore 64, a Commodore 128, an 80286 clone, and now an 80386 clone w/coprocessor, plus a large hard drive and megabytes of ram. I've programmed in Basic, 6502 and 8502 machine language, GWBasic, MicroSoft C, and finally Visual Basic.

Prior to the introduction of Windows 3.0, I had built up a point-and-shoot type menu system of my own creation which I was real proud of. After loading Windows for the first time, in two days my old menu program was history. This new graphics interface type approach to operating was beautiful. For the first time I got a glimmer of what maybe the Mac advocates had been talking about. Also, I wanted to program in the Windows environment. Part of the fun of programming is having the program items come up with the names that YOU want on them, located where YOU want them, and in the colors YOU want, and without warnings of impending legal doom if you don't register or license.

Then, at about the same time MicroSoft releases DOS 5.0, they also release a product called Visual Basic. I don't think MicroSoft realized how many programmers there are in the MS-DOS world, because I had a heck of a time getting my copy of Visual Basic. I put my name on a waiting list at EggHead in downtown Seattle, and it took about ten days to finally get my copy.

When I did get my copy I hurried home and did what all good programmers do, throw the disks in the drive, run install, and read the documentation later. The darn thing loaded with no problem, and we were off. The program comes with a super on screen tutorial, excellent on-line help, and some manuals that sort of follow the tutorial.

Creating a Windows program in Visual Basic is easy. When you first start, you are given a grid lined work area called a FORM. On the form you place OBJECTS, such as text boxes, command buttons, control bars, etc. You then assign PROPERTIES to these objects. Properties consist of colors, size, names locations, and so on. Whenever an object is suppose to do something, you call this action an EVENT. This is where programming comes in. You assign program code to do these events. In most cases not too much programming is required because Visual Basic has already provided routines for the most common actions. Each OBJECT has a built-in set of EVENT sub routines assigned to it.

I smiled to myself when I first saw the syntax of these sub routines. They looked remarkably like elements of a C structure. The only problem I encountered, (wouldn't have happened if I had read or used the tutorial first) was I wanted user input in a text box to be either entered by clicking the mouse button or pressing the enter key. Turned out I had used the "click ()" subroutine instead of the "keypress ()" subroutine. Seems the "keypress ()" subroutine will work with a click also.

After about 8 hours of programming over three days (half of which was wasted because I didn't read) I had written my first WINDOWS program. The user (my wife) inputs the balance remaining on our mortgage, a monthly payment, an interest rate, and the program tells her how many decades it will take to pay off the mortgage. She can then raise the monthly payment by 5 cents and see how many years are knocked off the mortgage. And all this is done in Windows.

I guess the bottom line is that the whole experience was just lots of fun. Moving objects around on the form, creating different sized buttons, experimenting with different events, is thoroughly enjoyable. If you've never programmed before, it will take you about 4 hours (if you read the manual) to write a really professional looking program. If you're an experienced programmer, it will take you about 8 hours to write a professional looking program. One final comment, you will learn more in 4 hours about how your computer works when you program, than you will in 2 years just using other people's applications.

Continue from page 6...A Skeleton Desktop Application:

```
        _DrawMenuBar    ;draw menu
bar
        rts
    ;.....................................
    ;
    ; InitDesktop data:
    ;
    MenuTable       anop  ;menu ptrs
        dc  i4'AppleMenu'
        dc  i4'FileMenu'
        dc  i4'EditMenu'
        dc  i4'SpecMenu'
        dc  i4'SampMenu'
    MTIndex equ *-MenuTable-4
    AppleMenu       dc
c'>>@\N1X',h'00'
        dc  c'—About Skeleton...\N256'
        dc  h'00'
        dc  c'—\N300D',h'00'
        dc  c'.'
    FileMenu dc     c'>> File
\N2',h'00'
        dc  c'—Open\N257*OoD',h'00'
        dc  c'—Close\N255*Ww',h'00'
        dc  c'—\N300D',h'00'
        dc  c'—Quit\N258*Qq',h'00'
        dc  c'.'
    EditMenu dc     c'>> Edit
\N3',h'00'
```

Continue to page 8

Continue from page 7Application:

```
        dc   c'—Undo\N250D*Zz',h'00'
        dc   c'—Cut\N251D*Xx',h'00'
        dc   c'—Copy\N252D*Cc',h'00'
        dc   c'—Paste\N253D*Vv',h'00'
        dc   c'—Clear\N254D',h'00'
        dc   c'.'

    SpecMenu        dc   c'>> Special
\N4',h'00'
        dc   c'—Boot
Slot...\N259D',h'00'
        dc   c'—Shut
Down...\N260',h'00'
        dc   c'.'

    SampMenu        dc   c'>> Samples
\N5',h'00'
        dc   c'—Window
1\N261*11',h'00'     ;***
        dc   c'—Window
2\N262*22',h'00'     ;***
        dc   c'—Sample
3\N263*33D',h'00'
        dc   c'.'
        end

    ********************************************
    *
    InitProgram start
    *
    * Initialize application specific data:
; * * *
    *
    ********************************************
        using        Globals

        stz  QuitFlag;clear flag
        stz  MLPtr  ;zero ML ptr
        stz  MLPtr+2
        rts
        end

    ********************************************
    *
    EventLoop       start
    *
    * Handle events: ;*** replace mod-
ule
    *
    ********************************************
        using        Globals

    ; Scan for events with TaskMaster:

        pha          ;result space
        PushWord #$FFFF  ;all events
        PushLong #TaskRec ;ptr to
record
        _TaskMaster
        PullWord EventType ;task code
        sta  TMCode ;task code

    ; Filter events passed to modeless
dialog:

        tax          ;test ML flags
        lda  MLFlagTbl,x
        and  #$FF
        beq  CheckEvent    ;don't tell ML

    ; Check for modeless dialog event:

        pha
        PushLong #TaskRec
        _IsDialogEvent
        pla          ;get Boolean
        beq  CheckEvent    ;not ML
event
        lda  #12     ;map ML event
        sta  EventType     ; to task #12

    ; Access event handler:

    CheckEvent lda   EventType
        asl  a
        tax
        jsr  (TaskTable,x)

    ; Check for continuation of event loop:

        lda  QuitFlag
        bpl  EventLoop     ;continue
scan
        rts  ;quit pgm
    ;.....................................
    ;
    ; EventLoop data:
    ;
    EventType ds     2       ;type of
event

    TaskTable anop
        anop ;GetNextEvent handlers
        dc   a2'DoNull'     ; 0 = nullEvt
        dc   a2'DoNothing' ; 1 =
mousDownEvt
        dc   a2'DoNothing' ; 2 =
mouseUpEvt
        dc   a2'DoNothing'  3 =
keyDownEvt
        dc   a2'DoNothing' ; 4 = unde-
fined
        dc   a2'DoNothing' ; 5 =
autoKeyEvt
        dc   a2'DoNothing' ; 6 =
updateEvt
        dc   a2'DoNothing' ; 7 = unde-
fined
        dc   a2'DoNothing' ; 8 =
activateEvt
        dc   a2'DoNothing' ; 9 =
switchEvt
        dc   a2'DoNothing' ;10 =
deskAccEvt
        dc   a2'DoNothing' ;11 =
driverEvt
        dc   a2'DoMLEvent';12 =
app1Evt ; * * *
        dc   a2'DoNothing' ;13 =
app2Evt
        dc   a2'DoNothing' ;14 =
app3Evt
        dc   a2'DoNothing' ;15 =
app4Evt
        anop ;TaskMaster handlers
        dc   a2'DoNothing' ;16 =
wInDesk
        dc   a2'DoMenu'    ;17 =
wInMenuBar
        dc   a2'DoNothing' ;18 =
wClickCalled
        dc   a2'DoNothing' ;19 =
wInContent
        dc   a2'DoNothing' ;20 =
wInDrag
        dc   a2'DoNothing' ;21 =
wInGrow
        dc   a2'DoClose'   ;22 =
wInGoAway
        dc   a2'DoNothing' ;23 =
wInZoom
        dc   a2'DoNothing' ;24 = wInInfo
        dc   a2'DoMenu'    ;25 =
wInSpecial
        dc   a2'DoNothing' ;26 =
wInDeskItem
        dc   a2'DoNothing' ;27 =
wInFrame
        dc   a2'DoNothing' ;28 =
wInactMenu
        dc   a2'DoNothing' 29 =
wClosedNDA
        dc   a2'DoNothing' ;30 =
wCalledSysEdit
        dc   a2'DoNothing' ;31 =
wTrackZoom
        dc   a2'DoNothing' ;32 =
wHitFrame

    MLFlagTbl anop   ;modeless event
flags:
        anop ;1=process, 0=bypass
        dc   i1'0,1,1,1,0,1,1,0,1,0'
;TM codes    0- 9
        dc   i1'0,0,0,0,0,0,0,0,0,1'
;        10-19
        dc   i1'0,0,0,0,0,1,0,0,0,0'
;        20-29
        dc   i1'0,0,0'      ;       30-32
      end

    ********************************************
    *
    DoMenu  start
```

```
*
* Handle menu events:
*
****************************************
        using   Globals

; Handle menu item (consecutive numbering):

        sec
        lda  TaskData
        sbc  #250  ;#250=0th  item
        asl  a
        tax
        jsr  (ItemTable,x)

; Unhighlight menu title:

        PushWord #0;unhilite
        PushWord TaskData+2 ;menu ID
        _HiliteMenu
        rts
;.....
;
; DoMenu data:
;
ItemTable anop   ;menu item handlers
        dc   a2'DoNothing' ;250 = Undo
        dc   a2'DoNothing' ;251 = Cut
        dc   a2'DoNothing' ;252 = Copy
        dc   a2'DoNothing' ;253 = Paste
        dc   a2'DoNothing' ;254 = Clear
        dc   a2'DoClose'   ;255 = Close
        dc   a2'DoAbout'   ;256 =
About...
        dc   a2'DoNothing' ;257 = Open
        dc   a2'DoQuit'    ;258 = Quit
        dc   a2'DoNothing' ;259 = Boot
Slot...
        dc   a2'DoShutDown'      ;260 =
Shut Down...
        dc   a2'DoWindow1'       ;261 =
Sample 1    ;***
        dc   a2'DoWindow2'       ;262 =
Sample 2    ;***
        dc   a2'DoNothing' ;263 =
Sample 3
        end

****************************************
*
DoQuit  start
*
* Quit this application:
*
****************************************
        using   Globals

        lda  #$8000
        sta  QuitFlag;set flag
        rts
        end

****************************************
*
DoNothing       start
*
* Ignore an event:
*
****************************************

        rts
        end

****************************************
*
DoAbout start
*
* Display alert window with credits:
*
****************************************

        pha             ;result space
        PushWord #0;C-string
        PushLong #0 ;no subst array
        PushLong #AboutStr ;ptr to
string
        _AlertWindow
        pla             ;button number
        rts
;.....
;
; DoAbout data:

AboutStr anop    ;alert string
        dc   c'0',i'53,150,137,490'
        dc   c'0/'
        dc   h'01',c'J',i'1'
        dc   h'01',c'S',i'1'
        dc   h'0D',c'Skeleton Desktop '
        dc   c'Application',h'0D'
        dc   h'01',c'S',i'0'
        dc   c'Version 1.0',2h'0D'
        dc   c'by Sandy Mossberg',h'0D'
        dc   c'(c) 1989 by Call -
A.P.P.L.E.'
        dc   c'/^#0'
        dc   h'00'
        end

****************************************
*
DoClose start
*
* Close front window: ;***  replace
module
*
****************************************
        using   Globals

        PushLong #0
        _FrontWindow            ;get
ptr to
        PullLong FrontWinPtr    ; front
wind
        lda  FrontWinPtr
        ora  FrontWinPtr+2
        beq  Done    ;no window

        PushLong FrontWinPtr    ;as-
sume NDA
        _CloseNDAbyWinPtr; window
        bcc  Done    ;NDA closed

        PushLong #0 ;get menu
        PushLong FrontWinPtr    ; item
ID in
        _GetWRefCon; wRefCon
        plx             ;ID in lo
        pla             ;discard hi

        phx             ;enable this
        _EnableMItem    ; menu item

        lda  FrontWinPtr
        cmp  MLPtr
        bne  ClosAppWind  ;not ML wind
        lda  FrontWinPtr+2
        cmp  MLPtr+2
        bne  ClosAppWind  ;not ML wind

        PushLong MLPtr          ;close
modeless
        _CloseDialog    ; dialog win-
dow
        jsr  CheckErr
        stz  MLPtr      ;zero
modeless
        stz  MLPtr+2            ; dia-
log ptr
        rts

    ClosAppWind PushLong FrontWinPtr
;close appl
        _CloseWindow            ; win-
dow

    Done    rts
        end

        copy tools.asm
        copy null.asm
        copy shutdown.asm
        copy windows.asm  ;***
```

Listing 2

```
;.....
;
; WINDOWS.ASM source listing (Part
4)
; by Sandy Mossberg
;.....
;
```

..continued to page 13

Micro Review of DOS 5.0

by Steve Cavanaugh

DOS 5.0 is out, and there's a review on every magazine cover. What better time to write one myself. At least that way, my own experiences will be documented.

Setup

DOS 5.0 comes with a setup program that is fully interactive. Setup prompts for everything, and forewarns that you will need 2.8 megs of disk space to make a full installation. The reason you need so much room is that setup makes a complete backup of your old version in a directory called old_dos.1. Setup also prepares a diskette called uninstall, from a blank diskette supplied by the user. If you aren't happy with the results of the installation, use the uninstall disk to put things back the way they were before installing DOS 5.0.

DOS 5 must be installed on drive C: for a hard drive installation, so if you have a 1 meg C: partition, you need to repartition before you can complete the installation. If you are too inexperienced to attempt repartitioning your drive, you are going to need some outside help. Another option would be to install to floppy disk, and start your PC from drive A: until you can get your hard drives rearranged.

When I ran the setup program, I was curious how it would react to my CONFIG.SYS since my PC had 4DOS installed. Setup politely removed my 4DOS shell command and replaced it with a plain vanilla shell command. Since I also use Disk Manager software for my 70 Meg hard drive, I was more than a little concerned about compatibility.

When I ran setup, the program immediately informed me that I was using an incompatible disk drive management program, and it instructed me to look in the Readme File on Disk 4, under a specific paragraph number.

The Readme File informed me that DOS 5 comes with an updated driver for the Disk Manager software, and to use the Expand command to un-archive it to my C: drive, then use the command Setup/u to restart the setup program. I followed instructions to the letter, and the setup procedure proceeded to completion. It wasn't until two days later that I discovered that the Disk Manager driver I had expanded still had a filename of DMDRVR.BI_, not DMDRVR.BIN. The instructions in the Readme File had failed to warn me of the potential for this problem. Oddly, DOS 5 didn't seem to care, although I think that's because all of my partitions are less than 32 megs. I deleted the old DMDRVR.BIN, renamed the new one, and I can't tell the difference.

Memory Management

DOS 5 comes with a new version of HIMEM.SYS and a driver called EMM386.SYS. My PC at home is a 386-20 with 1 meg of ram, so I can use all the memory help I can get. After the setup was completed, my HIMEM.SYS had been updated, and most of my configuration remained the same. I had gained some low DOS memory, going from about 527K bytes, to about 602K bytes.

I kept reading about UMB's in magazines, so I got out the manual (ACK!) and discovered that by adding the EMM386.SYS driver, I could load all of my device drivers and TSR programs into high DOS memory. I sent my mouse driver high, my DMDRVR.BIN went up there, my ANSI.SYS is loaded high, my scanner device driver is high, plus several other small device drivers. Also, DOS 5 loads most of itself high. After these changes, I had over 629K bytes free. Nice!

Then, came the shock. I run a lot of Windows software at home, and the first thing I tried, I got a "Not enough memory" error . . . Hmmmm. I checked under the Program Manager Help About command, and to my surprise Windows was running in Real Mode. I couldn't find anything in the manual on Windows 3 and memory management, so I logged on to Prodigy to see if I could find someone who knew the story. The story is - look in the Readme File on Disk 4. With EMM386.SYS loaded, Windows 3 will only run in Real or Enhanced Modes. So on my 1 meg 386, I'm stuck with Real Mode unless I want to move all my device drivers back to low DOS memory, or add more RAM. Sigh.

One new command I like a lot is the MEM command. If you just type MEM, you get about 6 lines of information on how much memory is available in each of the memory areas. If you type MEM/c, you get a much more detailed description of your memory, including where each program or device driver is loaded. This is one of the most descriptive displays of memory I've used. I can understand it.

One thing that continually bothers me is the lack of a list command. DOS 5 still supports the MORE pipe, but that's a real kludge compared to the 4DOS list command. The MEM/c command takes 2 screens (if you have your screen set to the standard 24 lines), so you have to keep piping the command through MORE.

Editor

DOS 5 comes with a new ASCII text editor called EDIT.COM. This reminds me of the text editor in Windows - small, simple, and easy to use. It's most redeeming feature is that it will be standard with every installation of DOS 5. No more EDLIN - hooray!

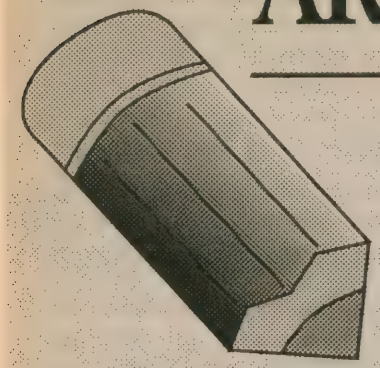
There is one thing about EDIT.COM that bothers me. The program must be placed in the same directory as QBASIC. At least, that was what the manual said. So just for fun I renamed QBASIC to QB. EDIT.COM would not start. It aborted with a message that it could not find QBASIC. Odd, especially since EDIT is a .COM file. I guess you won't find it running on an old PC under DOS 2.1.

DOS Shell

DOS 5 comes with a new program called DOS SHELL which is essentially an enhanced version of the file manager from PC TOOLS. You get a tree structure on the left side and files on the right. The big difference is you can move files by clicking and dragging them to the new location. You can also double click on a data file to call up the application and load the file automatically. If you have data files that DOS SHELL doesn't know about, you can set it up to recognize the file extension you're using, and load your program. One of the nice things about DOS SHELL is, with a VGA display, you can choose from about 6 different resolutions. You get between 25 and 60 lines of text, in several steps, in text or graphics modes. I set mine to 60 lines in graphics mode almost immediately. Then I put it back, because I had to squint to read some of the file names.

One nifty thing in DOS SHELL is a task swapper. The task swapper is designed to

ARNON LOAWHAKASETR



FREELANCE DESIGNER/ DESKTOP PUBLISHING SERVICES

Graphic Design/Illustrations/Production

Over 5 years of experience in Desktop Publishing & Graphic Design.

Specialized in books, manuals, newsletters and computer illustrations/maps. REASONABLE RATE.

FOR DESKTOP SUPPORT CALL: 223-4221

allow you to suspend a current program in favor of another program. The task switcher moves the current program to disk while you are loading and using the next program. I like the idea, especially since I had used a similar switcher on the Macintosh years ago. Unfortunately the task swapper crashes on my machine. I haven't called Microsoft yet, but if I run out of ideas, and nobody on Prodigy can help, I may have to resort to a 45 minute wait to get the answer.

DOSKEY Command

When I first decided to set up DOS 5, I did so with trepidation. I have been a satisfied user of DOS 3.3, running the 4DOS command processor for several years. I am very spoiled by 4DOS. It works the way I work, does the things I want, and I have no real need to change. Alas, sometimes an upgrade gives you things you didn't know you needed, so I went for it.

The trouble is, how to make DOS 5 act like 4DOS. In comes DOSKEY. DOSKEY is a new command that provides a command history buffer. You press the up-arrow key to recall the last command. DOS 5 let's you edit on the command line, so reissuing commands with minor variations is now fairly easy.

DOSKEY also allows you to invoke macros. DOSKEY macros are similar in nature to 4DOS aliases. You can issue any DOS command from a DOSKEY macro, and you can put multiple commands in a macro too. You can also put parameter substitutions in DOSKEY macros by using $1 through $9.

I wrote a DOSKEY macro to simulate the 4DOS MOVE command, by issuing a copy $1 to $2 and then a delete $1. Be careful doing this. If you enter a directory that doesn't exist, DOS will automatically think you want to copy to a file, and if your input parameter was a list such as *.*, all of your files will be deleted, and the last one will appear somewhere you don't expect, with a directory name as a file name.

QBasic

DOS 5 comes with QBASIC, which is a subset of Quick Basic. This takes the place of both GW-BASIC and BASICA. QBASIC comes with several games as examples of code.

Conclusion

After I get a memory upgrade, I'll be happy I switched to DOS 5. I will likely not use the editor or the DOS SHELL, but the memory improvement is probably worth the $39.95 I shelled out for it. There is a great deal more that I haven't touched upon. But really, if you need more detail than this, buy the program! I don't think you can go wrong here.

Bits and Bytes

by Bill Cowell

Odd Ball Computers

I still run across computer users who think the TI 99, Trash 80 (Tandy TR-80), or some other such vintage computer, is the greatest. They can't seem to understand why no one still supports their favorite. Well, I have just run across a large computer monthly which supports all these as well as the current state of the art machines. The publication is the "Computer Monthly". The price is only $1.95. It devotes a column to each of the obsolete computers and the want ad's also suggest the availability of products for some, if not all these relics. It also covers the MacIntosh.

M/S DOS 5.0

This has to be the hot topic of the summer. Microsoft has done an excellent job of testing this new operating system, but there are still some problems. They tried to anticipate as many of these problems as possible, as you can tell by reading their 13 page ReadMe File. If you want a real education all you have to do is check out one of the BBS's or even Prodigy's Computer Club. There are a number of little goodies in DOS 5.0, but the ones most users will like are the availability of additional base memory, the Windows like shell, and on-line help. For new PC users the shell offers a much friendlier user interface than the enigmatic DOS prompt. It provides point and click file management and program execution capability. It also provides a task switcher similar to the one in Software Carousel. The task switcher won't support cut and paste between programs, like you get under Windows, but it will allow you to quickly switch between programs.

On a AT or a 386 machine, with more than one MB of memory, DOS 5.0 will move about 40-45k of the operating system into upper memory. No such benefit with an XT, you could actually end up with less working memory than you now have with DOS 3.3. For XT owners I recommend that you stay with DOS 3.3. If you don't already have DOS 3.3, it should be available now at a bargain price. Find someone that is upgrading to DOS 5. It may even be a bargain from some of the suppliers as they upgrade to DOS 5.0.

One caution worth mentioning, relating to defragmentors, is older versions of these essential hard disk clean up utilities can trash hard drives when you are operating under DOS 5.0. Newer versions of these programs shouldn't have this problem. The problem should only occur if you interrupt the defragmenting in the middle of the process. If you are now using DOS 5.0, it would be worth a call to the developer of your favorite defragmentor to see if there is a problem with your version. One upgrade that is sure to do the trick is version 7 of PC Tools. Central Point worked closely with Microsoft during the development of DOS 5.0, so the latest PCT should be compatible with it.

The one problem that I have run into when installing 5.0, is with third-party hard disk drivers. Disk Manager is the program that gave me some trouble. It required a little extra effort to get DOS 5 into to a customer's 386 machine. We ran the Expand program that comes with the DOS, and then used the Setup/U command to install the DOS with no further difficulties.

Windows 3.1

When Windows 3.1 is released, maybe by the end of the year, it will offer several improvements. The most important will be those that are suppose to improve its speed. One thing they are doing to increase speed is to use the latest in FastDisk technology. This will provide a virtual device for direct communication with the hard drive controller, bypassing both DOS and the BIOS. Some people feel that this approach may create a problem with some disk controllers and third party disk partitioning software. We will just have to wait and see. Also included in the update will be the TrueType font technology. Hopefully, this will improve printer control under Windows. TrueType is MicroSoft's and Apple's answer to Adobe Post Script. Beta test copies of the program included 13 scalable fonts including Arial, Times, New Roman, Symbol, and Courier. Another welcomed improvement will be the changes to the File Manager. It will show the directories on the left, and the files in the highlighted directory, on the right. This will be much like many of the third party file managers, such as the one in PC Tools. The program will also support additional memory and the special requirements of laptop computers.

Shareware of the Month

This is actually a free program from Qualitas. Qualitas makes some of the better memory management software. They are using this free Program to introduce users to their products. The program is called ASQ. It includes a memory management tutorial and will also perform an analysis of your system. It does a very thorough analysis of what your current hardware and software configuration is and makes recommendations for improvements. If all you used it for was as a tutorial on memory, it's more than worth a little time spent with it.

Modems and Fax Boards

I have been holding off purchasing a FAX board, because I expected the vendors to develop a board that combines modem and fax functions. I was also hoping for a lower price. Well, they have now done both. The following are a couple of new units that offer both capabilities. The first was the TwinCom 24/96 from Flohr and Image Communication. Their unit has 9600 fax send and receive capability and includes a 2400 baud modem. The unit comes complete with software. List price is only $149. Actually, this unit has been out since November of last year. The developer had trouble at first, because no one could believe that they could get a reliable unit from an unknown vendor. This has been overcome, and they are now shipping large quantities of the units. The units are assembled in the U.S. using a fully automated surface mount assembly facility. They are pumping them off the line at one every two minutes. The automated line and the use of the dual function Rockwell chip is the reason for the low price. Similar units are now available from several company's including Cardinal Technologies and Practical Peripherals. List price of the Cardinal unit is $199. I am leaning toward the Cardinal unit, since my experience with their earlier 2400 baud modems has been good. The Practical Peripherals units would also be a good option. Their claim to fame is their five year warranty. The Practical Peripherals units have a list price of $209.

Video Boards

Video boards are another item which have recently seen a big increase in capability and lower prices. One such board is the Cardinal VGA732 VGA graphics card. This board incorporates the RAMDAC chip. With this chip the board is capable of generating over 32,000 colors at a resolution of 800 x 600.

MAC Notes

Reference the above paragraph on Fax/Modems. Ventel makes a miniature 2400 baud modem/ 9600 send Fax unit that is now available in a Macintosh version. List price is $189.

Editor's Note: Bill Cowell is the owner of Cowell Computers, (206-242-3692)

continued from page 9

```
    ; Consult Listing 1 for changes to
    ;SKELETON.ASM and NULL.ASM
    ;
    ********************************************
    *
    DoWindow1   start
    *
    * Put an application window on the
Desktop:
    *
    ********************************************
        using       Globals

        lda  askData ;put item ID
        sta  W1RC    ; in wRefCon

        PushLong #0         ;create win-
dow
        PushLong #Wind1Rec
        _NewWindow
        jsr  CheckErr
        PullLong Wind1Ptr  ;save wind
ptr
        PushWord TaskData ;disable this
        _DisableMItem      ; menu item
        rts
    ;.....
    ;
    ; DoWindow1 data:
    ;
    Wind1Ptr ds      4      ;window 1
ptr

    Wind1Rec anop    ;window 1 parmlist
        dc   i2'Wind1End-Wind1Rec'
;table size
        dc   i2'%1101110110100100'
frame flags
        dc   i4'Wind1Title' ;title string
ptr
    W1RC    dc       i4'0'  ;application
refCon
        dc   i2'0,0,0,0'    ;zoom size
        dc   i4'W1ColTbl'  ;color table
ptr
        dc   i2'0,0' ;content/data offset
        dc   i2'250,600'   ;data height/
width
        dc   i2'0,0' ;grow height/width
        dc   i2'9,12'       ;arrw scroll
vert/horiz
        dc   i2'141,308'   ;page scroll
vert/horiz
        dc   i4'0'    ;info bar refCon
        dc   i2'0'    ;info bar height
        dc   i4'0'    ;frame defProc ptr
        dc   i4'0'    ;info bar defProc
ptr
        dc   i4'Wind1Update'      ;up-
date defProc ptr
        dc   i2'30,110,180,430' ;coor-
dinates
        dc   i4'-1'   ;plane (front=-1)
        dc   i4'0'    memory ptr
    Wind1End anop

    Wind1Title  str   ' Application Win-
dow '

    W1ColTbl anop    ;window 1 color
table
        dc   2'$0000'      ;frame
        dc   2'$0F00'      ;title/inact
title bar
        dc   2'$020F'      ;active title
bar
        dc   2'$00F0'      ;grow box
middle outline
        dc   2'$00F0'      ;info bar in-
side outline
        end

    ********************************************
    *
    Wind1Update start
    *
    * Window 1 content definition proce-
dure:
    *
    ********************************************

        phb
        phk
        plb

        PushWord #20       ;position pen
        PushWord #20
        _MoveTo

        PushLong #Wind1Str ;draw win-
dow
        _DrawString ; contents

        plb
        rtl                ;note RTL
    ;.....
    ;
    ; Wind1Update data:
    ;
    Wind1Str str      'Play with me...'
             end

    ********************************************
    *
    DoWindow2       start
    *
    * Put a modeless dialog on the Desk-
top:
    *
    ********************************************
        using       Globals

    ; Create modeless dialog window:

        PushLong    # 0          ;result
space
        PushLong    #MLRect
;coords ptr
        PushLong    #MLTitle    ;title
ptr
        PushLong    #-1  ;plane
        PushWord    #$C0A0      ;frame
flags
        PushLong    TaskData    ;applic
refCon
        PushLong    # 0         ;zoom
ptr
        _NewModelessDialog
        jsr  CheckErr
        PullLong MLPtr          ;dialog
ptr

    ; Add contents to modeless dialog
window:

        PushLong MLPtr          ;ptr to
Okay
        PushLong #OkButn  ; button
        _GetNewDItem

        PushLong MLPtr          ;ptr to
Cancel
        PushLong #CancButn ; button
        _GetNewDItem

        PushLong MLPtr          ;ptr to
edit
        PushLong #EdTitle  ; line title
        _GetNewDItem

        PushLong MLPtr          ;ptr to
edit
        PushLong #EdBox    ; line box
        _GetNewDItem

    ; Disable this menu item:

        PushWord TaskData
        _DisableMItem
        rts
    ;.....
    ;
    ; DoWindow2 data:
    ;
    MLRect  dc       i2'80,30,150,400'
    MLTitle str      ' Modeless Dialog
Window'
    TxtOk   str      'Okay'
    TxtCanc str      'Cancel'
    TxtBox  str      'Type Name:'
    TxtNull str      ''
```

Continue to page 14

..continued from page 13

```
OkButn   anop          ;OK button template
         dc   i2'45,200,60,280'   ;coordinates
         dc   i2'$000A'    ;type (button)
         dc   i4'TxtOk'    ;descriptor ptr
         dc   i2'0'   ;value
         dc   i2'3'   ;bit flag
         dc   i4'0'   ;color table ptr

CancButn anop          ;Clear button template
         dc   i2'2'   ;item ID
         dc   i2'45,80,60,160'    ;coordinates
         dc   i2'$000A'    ;type (button)
         dc   i4'TxtCanc' ;descriptor ptr
         dc   i2'0'   ;value
         dc   i2'2'   ;bit flag
         dc   i4'0'   ;color table ptr

EdTitle  anop          ;edit title template
         dc   i2'3'   ;item ID
         dc   i2'12,10,22,90'     ;coordinates
         dc   i2'$800F'    ;type (statText+
;                  ;itemDisable)
         dc   i4'TxtBox'   ;descriptor ptr
         dc   i2'0'   ;value
         dc   i2'0'   ;bit flag
         dc   i4'0'   ;color table ptr

EdBox    anop    ;edit line template
         dc   i2'4'          ;item ID
         dc   i2'10,100,25,350'   ;coordinates
         dc   i2'$0011'    ;type (editLine)
         dc   i4'TxtNull'  ;descriptor ptr
         dc   i2'64'  ;value
         dc   i2'0'          ;bit flag
         dc   i4'0'          ;color table ptr
         end

*********************************************
*
DoMLEvent      start
*
* Handle events in modeless dialog:
*
*********************************************
         using     Globals

; Check activity of modeless dialog:

         lda       MLPtr
         ora       MLPtr+2
         beq       Done      ;MLdlog inact

; Check for and handle special menu items:

         lda       TMCode
         cmp       #25
         bne       DoMLEvt1  ;not wInSpecial

         PushWord  #0 ;unhighlight
         PushWord  TaskData+2 ; menu title
         _HiliteMenu

         lda       TaskData
         cmp       #255
         bne       DoMLEvt1  ;not Close item
         jmp       DoClose   ;close dlog box

; Handle modeless dialog event:

DoMLEvt1 pha ;result space
         PushLong  #TaskRec  ;event rec ptr
         PushLong  #MLPort   ;store port ptr
         PushLong  #ItemHit  ;item hit ptr
         _DialogSelect       ;check for hit
         pla                 ;get Boolean

         beq       Done      ;no item hit
         lda       ItemHit   ;get item ID
         cmp       #3
         bcs       Done      ;button not hit
         _SysBeep            ;OK/Cancel hit
Done     rts
;.....
;
; DoMLEvent data:
;
MLPort   ds   4    ;ptr to dialog's GrafPort
ItemHit  ds   2    ;ID of item hit
         end
```

EDITOR'S NOTE:

Permission was granted to TechAlliance to print Part 4 of this article written by Sandy Mossberg. Part 1 appeared in the July '89 issue of Call A.P.P.L.E.; Part 2 appeared in the Autumn '89 issue of Call A.P.P.L.E.; Part 3 appeared in the Winter '90 issue of Call A.P.P.L.E.

These articles may not be reprinted without the written permission of the author.

In addition, Cecil Fretwell, TechAlliance's former Technical Editor, has converted the Assembly Language version to "C". The "C" version is available on disk for $20.00. Cecil may be contacted at the following address:

Cecil Fretwell
2605 Highview Avenue
Waterloo, IA 50702
(319) 236-0961 Central Time Zone

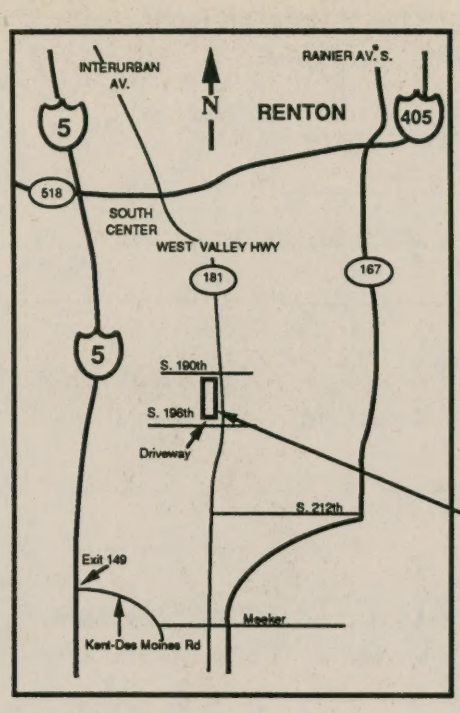
Norm Dodge has a Merlin translation of the above article. He may be contacted at the following address:

Norm Dodge
1920 4th Avenue North
Seattle, WA 98109
(206) 285-6676 Pacific Time Zone

VOLUNTEER NEWSLETTER POSITIONS AVAILABLE

- **Staff Reporters for each Special Interest Group**
- **Contributing Writers**
- **Co-editor**
- **Final Proofreader**
- **Distribution Volunteers**
- **Advertising Manager**

Contact the TechAlliance Office if you are interested in any of the above volunteer positions. Being a volunteer for your club is a rewarding and fun experience!



TechAlliance

. . .a computer cooperative

Important Numbers

Office: 206/251-5222
BBS: 206/251-0543
206/251-6784 (9600 Baud)
FAX: 206/251-6190

Office Hours: 9:30 - 3:00 pm

All meetings held at:

19309 W. Valley Hwy
Suite R109
Kent, WA 98032

People Helping People with Personal Technology

August 1991

Special Interest Group Meeting Information

APPLE // Family

Apple // SIG has rescheduled Ken Franklin, Reliefware author, to demonstrate his games. He was previously scheduled to do a demonstration in March of this year, but was called to duty in Saudi Arabia. Ken is a very special programmer because he donates the money from his programs to help families in need, thus the name Reliefware. InCider A+ Magazine, August '91 issue, under the Editor's Choice Column gave Ken's shareware/Reliefware program, MILESTONES 2000, a very favorable review. Among his other programs are ONE-ARM BATTLE and PLUNDER .So come join us at the August meeting for what promises to be a very enjoyable evening.

Macintosh SIG

No-Host Pizza & Salad Party

The August Meeting of the TechAlliance Macintosh Special Interest Group (MacSIG) will have no formal agenda. However, there will be a meeting: Pizza is the only scheduled activity (there is a very good chance that at least one Mac will be there, as well, for those who couldn't justify a Mac group without one; "public domain" software will be available).

7:00 p.m.
Godfather's Pizza
16800 Southcenter Parkway
tel. 575-4008

MS-DOS

The MS-DOS SIG meeting has been cancelled until September. But plan on an exciting meeting then. We have tentatively scheduled a meeting on MIDI, so it should be an exciting. Please be sure to watch the newsletter for the details next month. Also, we will have the MS-DOS Public Domain and Shareware library available at the meeting. We have the best prices in town. And for an even better bargain, bring your own disks! We will copy as many as possible for you at the meeting. Enjoy your summer and see you in the fall.

Mac Programmers SIG

Expect our usual round table discussion of programming issues and topics. If the need arises, we will break up into smaller groups for more detailed discussions.

TechAlliance Application for Membership or Renewal

Yes! Sign me up for one year's membership in TechAlliance for $25.
(International memberships are $35.)

Name _____ Daytime Phone _____

Address _____ Home Phone _____

_____

City _____ State _____ Zip _____

☐ Check Enclosed ☐ New Membership

☐ Charge my bank card (VISA or MasterCard) ☐ Renewal ☐ I will be using the BBS

Card Number _____ Expiration Date _____

Signature _____

CS0891

AUGUST Meeting Schedule

***Advance sign-up required for these meetings.*

Tue., August 6th:	IIGS Beginners Session - 7:00 p.m. IIGS (& II Family) SIG - 7:30 p.m.	Wed., August 21st:	Mac Programmers SIG - 7:00 p.m. MS-DOS SIG meeting canceled.
Tue., August 13th:	Mac Help Session - 6:45 p.m. Mac SIG meeting - 7:30 p.m.		Meetings will resume in September
Wed., Augurst 14th:	Amiga New User Class - 6:30 p.m.** AMIGAlliance - 7:30 p.m.	Tue., Sept 3rd:	IIGS Beginners Session - 7:00 p.m.IIGS (& II Family) SIG - 7:30 p.m.
Sat., August 17th:	Amiga New User Class - 10:00 a.m.**		
Tue., August 20th:	Voluntary Advisory Comm. 7:00 p.m.	***Meeting Location:***	All meetings are held at the office unless otherwise specified.

TechAlliance
Suite R-109
19309 68th Ave S.
Kent, WA 98032

Bulk Rate
U.S. Postage
PAID
Seattle, Washington
Permit No. 2726